



Simpson, R.N., and Liu, Z. (2016) Acceleration of isogeometric boundary element analysis through a black-box fast multipole method. *Engineering Analysis with Boundary Elements*, 6, pp. 168-182.

There may be differences between this version and the published version. You are advised to consult the publisher's version if you wish to cite from it.

<http://eprints.gla.ac.uk/117255/>

Deposited on: 17 May 2016

Enlighten – Research publications by members of the University of Glasgow  
<http://eprints.gla.ac.uk>

# Acceleration of isogeometric boundary element analysis through a black-box fast multipole method

R. N. Simpson<sup>\*,a</sup>, Z. Liu<sup>a</sup>

<sup>a</sup>*School of Engineering, University of Glasgow, Glasgow, G12 8QQ, U.K.*

---

## Abstract

This work outlines the use of a black-box fast multipole method to accelerate the far-field computations in an isogeometric boundary element method. The present approach makes use of T-splines to discretise both the geometry and analysis fields allowing a direct integration of CAD and analysis technologies. A black-box fast multipole method of  $O(N)$  complexity is adopted that minimises refactoring of existing boundary element codes and facilitates the use of different kernels. This paper outlines an algorithm for implementing the open-source black-box fast multipole method BBFMM3D<sup>1</sup> within an existing isogeometric boundary element solver, but the approach is general in nature and can be applied to any boundary element surface discretisation. The  $O(N)$  behaviour of the approach is validated and compared against a standard direct solver. Finally, the ability to model large models of arbitrary geometric complexity directly from CAD models is demonstrated for potential problems.

*Key words:* isogeometric analysis, T-splines, boundary element method, black-box fast multipole method

---

## 1. Introduction

In the majority of modern industrial engineering and design workflows Computer Aided Design (CAD) and analysis software play a crucial role in reducing the overall design lifecycle. The iterative nature of design requires tight integration of CAD and

---

<sup>\*</sup>Corresponding author

*Email address:* robert.simpson.2@glasgow.ac.uk (R. N. Simpson)

<sup>1</sup><https://github.com/ruoxi-wang/BBFMM3D>

analysis software, but modern workflows are inhibited by cumbersome fixing and de-featuring algorithms that must be used in the transition from CAD models to analysis models. The disparity between CAD and analysis is one of the biggest challenges facing engineering design which has inspired research into new discretisation approaches that unify or greatly ease the transition from CAD to analysis and vice versa.

One of the most active research areas that aims to address the disparity between CAD and analysis is the field of isogeometric analysis (IGA) [28] that uses spline-based discretisations generated by CAD software as a basis for analysis thus providing a framework that unifies CAD and analysis. Since the seminal paper of [28], the concept has expanded rapidly into several applications including acoustics [47, 38], vibrations [15], elasticity [1, 42], electromagnetics [49, 10] and fluid flow [3, 17, 26]. Early work on IGA has focussed on the use of Non-Uniform Rational B-Splines (NURBS) [39] due to their popularity within modern commercial CAD software, but limitations stemming from their tensor-product nature have prompted research into alternative CAD discretisations including subdivision surfaces [13, 14], PHT splines [35, 50], LR B-splines [29], T-splines [2] and THCCS [52]. From a commercial perspective, the two technologies which have made the largest impact include subdivision surfaces and T-splines. Subdivision surfaces are ubiquitous within the computer animation industry but at present, they have yet to penetrate the CAD software market. T-splines offer a promising route to overcome the tensor product nature of NURBS while also providing backwards compatibility with existing NURBS technology. From an analysis perspective, T-splines have opened up interesting routes for integrated design and analysis technologies through properties such as water-tight geometries and local refinement algorithms. T-splines were first used in an analysis context in [2] and subsequently analysis-suitable T-splines [32, 9] were proposed that satisfy important analysis properties while retaining flexible geometry and modification algorithms. Further research includes efficient evaluation of T-spline basis functions through Bézier extraction [43].

A popular approach in CAD is to represent geometry in terms of a surface or Boundary-Representation (B-Rep) through appropriate geometry discretisations such as connected NURBS patches, T-spline or subdivision surfaces. Such surface discretisations are insufficient for volumetric analysis methods such as the finite element

method but provide the necessary data structures for analysis methods based on surfaces such as shell and boundary integral formulations. The limitations of boundary integral approaches are well-known, but assuming the use of such an approach is valid, they are found to be a particularly attractive approach for integrated design and analysis. By adopting a common discretisation for both geometry and analysis, isogeometric boundary element methods completely circumvent meshing procedures and eliminate geometry error promoting design software that truly integrates CAD and analysis. The idea has been explored in the context of several applications including elastostatics [46, 45, 51], shape optimization [7, 19, 31] acoustics [47, 37, 38] and underground excavations [6].

A well-known feature of the BEM approach is the debilitating  $O(N^2)$  asymptotic behaviour for matrix assembly and  $O(N^3)$  behaviour of direct solvers that eventually dominates for large problems. For practical engineering problems this manifests itself as large runtimes and heavy memory demands that often completely prohibit the use of direct solvers. Instead, matrix compression techniques which reduce the overall solver complexity to  $O(N \log N)$  or  $O(N)$  must be used. At present, the most popular techniques include: the Fast Multipole Method (FMM) [21, 11, 12, 36] and Hierarchical (H-) matrices [24, 22, 23] which make use of low-rank compression methods such as Adaptive Cross Approximation (ACA) [5, 30]. These techniques are all based on the same fundamental concept of approximating the smooth nature of the kernel for far-field interactions through efficient hierarchical data structures that allow for fast matrix-vector computations within an iterative solver. More recent research has focussed on the development of fast direct solvers (e.g. [20, 8]) that have shown advantageous properties over iterative techniques and offer a promising direction for future BEM solvers.

From an implementation standpoint, preference is often given to ACA and H-matrix methods which perform matrix compression in a purely algebraic manner, in contrast to the majority of FMM implementations which require extensive changes to BEM software. However, there exist black-box FMM implementations that overcome these limitations [53, 18, 33] opening up efficient  $O(N)$  FMM algorithms to BEM software. The present paper is based on such techniques.



Previous work on accelerating isogeometric BEM computations includes FMM compression for 2D Laplace problems [48], H-matrices to accelerate 2D and 3D elasticity applications [34] and a comparative study of Wavelet, FMM and ACA compression defined over parametric surfaces [25]. All of these studies have made use of tensor product parameterisations in the form of NURBS or rational Bézier surfaces.

The present paper outlines an approach for accelerating BEM computations in the framework of isogeometric analysis by employing a black-box FMM and adopting T-splines to discretise both the surface geometry and analysis fields. A collocation approach is chosen in the present study, but the techniques are applicable also to Galerkin and Nyström methods. Through the use of a black box FMM algorithm, the changes required to any existing BEM code are kept to a minimum. The use of T-splines allows direct integration of computational geometry and analysis technology while overcoming the inherent limitations of tensor product surfaces. The combination of these technologies offers a significant step forward towards integrated design and analysis for industrial applications.

The paper is organised as follows: a brief overview of the black-box FMM algorithm is given highlighting common FMM terminology and its relation to traditional BEM notation; the boundary element discretisation procedure that allows a system of equations to be formed is stated; an overview of T-spline discretisation technology is described; the algorithm for computing the matrix-vector product through the black-box FMM for fast BEM solve times and reduced memory consumption is detailed and finally, numerical examples are given to verify the implementation and assess its asymptotic behaviour against a standard direct solver for potential problems. All algorithms and numerical examples in the present work are based on three-dimensional problems.

## **2. Fast multipole methods**

Fast multipole methods were originally developed to overcome the intractable computational complexity of N-body problems when solved by direct means. Such prob-

lems can be expressed as

$$f(\mathbf{x}_i) = \sum_{j=1}^{N_s} K(\mathbf{x}_i, \mathbf{y}_j) \sigma_j \quad i = 1, 2, \dots, N_f \quad (1)$$

where  $f(\mathbf{x}_i)$  is the desired force or field,  $K(\mathbf{x}, \mathbf{y})$  is a problem specific kernel,  $\{\sigma_j\}_{j=1}^{N_s}$  is a set of charges,  $\{\mathbf{x}_i\}_{i=1}^{N_f}$  a set of field points and  $\{\mathbf{y}_j\}_{j=1}^{N_s}$  a set of source points. In the case  $N_s = N_f = N$  and (1) is applied directly, the computation time scales as  $O(N^2)$  which necessitates acceleration methods for large problems. Early work on the FMM applied to three-dimensional problems formulated methods that scale as  $O(N \log N)$  with subsequent improvements in algorithms achieving scaling of  $O(N)$ . Many variants of the FMM exist, but all are based on the same fundamental algorithm:

1. **Prescribed tolerance:** a tolerance  $\epsilon$  is prescribed to determine the number of terms retained in far-field expansions.
2. **Subdivision of space:** a hierarchical subdivision of space is constructed consisting of  $m$  levels indexed by  $k = 0, 1, \dots, m$ . Octree subdivision is commonly used for three-dimensional problems. For each level of the octree, a set of cells  $\{\mathcal{Y}_a^k\}_{a=1}^{n_{cell}^k}$  is defined. Source and field points are assigned to cells in every level.
3. **Upwards pass:** far-field expansions are computed for each cell at the lowest level  $m$  of the tree. Far field expansions for cells in level  $m - 1$  and higher are computed from expansions in lower levels through a Moment-to-Moment (M2M) translation operator.
4. **Downwards pass:** working down the tree, local expansions are formed for each cell  $\mathcal{Y}_a^k$ . These are calculated through a Moment-to-Local (M2L) operator for cells in the interaction list<sup>2</sup> of  $\mathcal{Y}_a^k$  and a Local-to-Local (L2L) operator applied to the parent cell of  $\mathcal{Y}_a^k$ .
5. **Evaluation:** working at the lowest level of the tree, the FMM approximation of  $f(\mathbf{x}_i)$  is computed by finding the cell at the lowest level of the tree which contains  $\mathbf{x}_i$ . The local expansion of this cell is used to compute the far-field

---

<sup>2</sup>See [4] for a thorough definition of FMM terms including well-separated, near-neighbours and the interaction list of a cell.

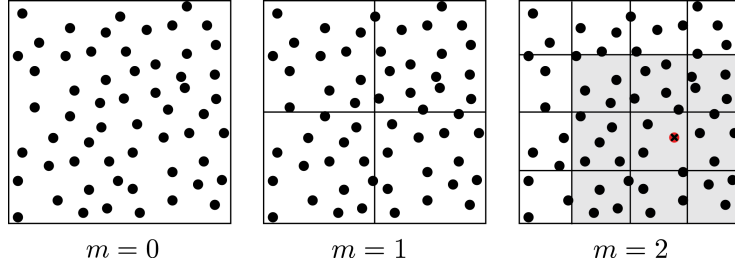


Figure 1: Illustration of increasing levels of refinement illustrated through quadtree subdivision. The near-neighbour cells for a field point (denoted by  $x$ ) located in a cell at  $m = 2$  are shaded in grey.

approximation with the near-field computed directly by summing over all near-neighbours.

### 2.1. Black-box fast multipole method

In the case of the black-box algorithm of [18], far-field expansions are based on Chebyshev interpolation and M2L operators are constructed through reduced rank operators calculated by Singular Value Decomposition (SVD). A particularly beneficial feature of this approach is its ability to handle arbitrary kernels in contrast to conventional FMM implementations that often require significant code rewrites for alternative kernels. This justifies the use of such an approach in the present study.

To accelerate N-body computations using the black-box code of [18], the following specific inputs are required:

1. **Tolerance parameters:** consisting of the target precision  $\epsilon$  used to compute SVD cutoff parameters and  $n_{ch}$ , the number of Chebyshev nodes used to interpolate in each coordinate direction.
2. **Hierarchical subdivision parameters:** comprising of  $m$ , the number of levels in the tree hierarchy and  $L$ , the side-length of the smallest cube enclosing the domain.
3. **Kernel,  $K(x, y)$ :** prescribed either analytically or numerically.
4. **Coordinates:** consisting of the set of field points  $\{x_i\}$  and source points  $\{y_j\}$ .
5. **Source charges:** denoted by the set  $\{\sigma_j\}$ .

In the case of a BEM formulation, further machinery is required before the black-box algorithm can be used to accelerate far-field computations. The following section outlines how a BEM discretisation can be recast in the context of N-body simulations making use of T-splines as an ansatz for both the geometry and analysis fields.

### 3. BEM discretisation

Before the formulation for the mixed boundary value problem is given, we first state the Dirichlet and Neumann interior boundary value problems that are used in its construction. The reader may wish to consult [41] for definitions of relevant trace spaces. We assume the problem is prescribed over a domain  $\Omega$  with a Lipschitz boundary  $\Gamma := \partial\Omega$ . The semi-discrete boundary element formulation for the Dirichlet boundary value problem is stated as: given boundary data  $g_D \in H^{1/2}(\Gamma)$  and a set of collocation points  $\{\mathbf{x}_I\}_{I=1}^{n_c}$ , find  $u_N \in H^{-1/2}(\Gamma)$  such that

$$(Vu_N)(\mathbf{x}_I) = \frac{1}{2}g_D(\mathbf{x}_I) + (Kg_D)(\mathbf{x}_I) \quad (2)$$

where the operators  $V$  and  $K$  are defined as

$$(Vu)(\mathbf{x}) := \int_{\Gamma} G(\mathbf{x}, \mathbf{y})u(\mathbf{y}) \, d\Gamma(\mathbf{y}) \quad (Ku)(\mathbf{x}) := \int_{\Gamma} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} u(\mathbf{y}) \, d\Gamma(\mathbf{y}) \quad (3)$$

and  $G(\mathbf{x}, \mathbf{y})$  denotes the relevant Green's function. The factor of  $1/2$  in (2) assumes that all collocation points lie on a smooth portion of the boundary. Likewise, the Neumann problem is stated as: given boundary data  $g_N \in H^{-1/2}(\Gamma)$ , find  $u_D \in H^{1/2}(\Gamma)$  such that

$$\frac{1}{2}u_D(\mathbf{x}_I) + (Ku_D)(\mathbf{x}_I) = (Vg_N)(\mathbf{x}_I). \quad (4)$$

In the case of a mixed-value problem the boundary is partitioned into Dirichlet and Neumann boundaries  $\Gamma_D$  and  $\Gamma_N$  respectively such that  $\Gamma = \Gamma_D \cup \Gamma_N$ ,  $\Gamma_D \cap \Gamma_N = \emptyset$ . The problem now becomes: given boundary data  $g_D \in H^{1/2}(\Gamma_D)$ ,  $g_N \in H^{-1/2}(\Gamma_N)$  find  $u_N \in H^{-1/2}(\Gamma_D)$ ,  $u_D \in H^{1/2}(\Gamma_N)$  such that (2) holds  $\forall \mathbf{y} \in \Gamma_D$  and (3) holds  $\forall \mathbf{y} \in \Gamma_N$ .

To arrive at a fully discrete formulation, the unknown fields  $u_D, u_N$  are discretised as

$$u_D(\mathbf{y}) = \sum_{J=1}^{n_d} \phi_J^D N_J(\mathbf{y}) \quad u_N(\mathbf{y}) = \sum_{J=1}^{n_n} \phi_J^N \hat{N}_J(\mathbf{y}) \quad (5)$$

in which  $\{\phi_J^D\}_{J=1}^{n_d}$ ,  $\{\phi_J^N\}_{J=1}^{n_n}$  are sets of unknown Dirichlet and Neumann nodal coefficients respectively and  $\{N_J\}_{J=1}^{n_d}$ ,  $\{\hat{N}_J\}_{J=1}^{n_n}$  are sets of continuous and discontinuous basis functions respectively. The prescribed data  $g_D$ ,  $g_N$  can be discretised in an analogous manner.

Substituting expressions (5) into (2) and (4) and including the discretised boundary data, a system of equations is formed as

$$\left[ \begin{array}{c|c} \frac{1}{2}\mathbf{I} + \mathbf{K}_D & \mathbf{V}_N \end{array} \right] \begin{bmatrix} \phi_D \\ \phi_N \end{bmatrix} = \left[ \begin{array}{c|c} \mathbf{V}_G & \frac{1}{2}\mathbf{I} + \mathbf{K}_G \end{array} \right] \begin{bmatrix} \mathbf{g}_N \\ \mathbf{g}_D \end{bmatrix} \quad (6)$$

where  $\phi_D$ ,  $\phi_N$  are vectors of unknown nodal Dirichlet and Neumann coefficients,  $\mathbf{g}_D$ ,  $\mathbf{g}_N$  are vectors of known nodal coefficients,  $\mathbf{I}$  is the identity matrix and the components of  $\mathbf{K}_D$  and  $\mathbf{V}_N$  are given by

$$[K_D]_{IJ} = \int_{\Gamma_N} \frac{\partial G(\mathbf{x}_I, \mathbf{y})}{\partial n} N_J(\mathbf{y}) d\Gamma(\mathbf{y}) \quad I = 1, 2, \dots, n_c \quad J = 1, 2, \dots, n_d \quad (7)$$

and

$$[V_N]_{IJ} = \int_{\Gamma_D} G(\mathbf{x}_I, \mathbf{y}) \hat{N}_J(\mathbf{y}) d\Gamma(\mathbf{y}) \quad I = 1, 2, \dots, n_c \quad J = 1, 2, \dots, n_n. \quad (8)$$

The components of  $\mathbf{K}_G$  and  $\mathbf{V}_G$  are given by similar expressions. Performing the matrix-vector multiplication on the right hand side of (6) and collecting terms on the left hand side, the system of equations can be written as

$$\mathbf{K}\phi = \mathbf{f} \quad (9)$$

which is in a form that can now be solved.

### 3.1. T-spline basis

In the present study, a T-spline basis is chosen to discretise expressions (5) and to provide a discretisation of the surface geometry. From a practical perspective this provides significant advantages over conventional discretisation or meshing procedures since a T-spline basis can be generated automatically by CAD software and used directly for analysis. This has important implications for design workflows where the creation of analysis models is time-consuming and expensive.

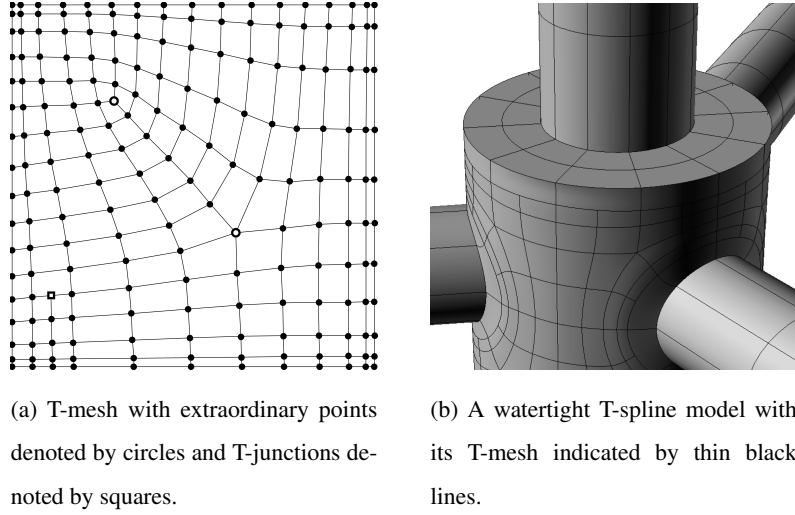


Figure 2: Unstructured T-meshes

A T-spline discretisation is defined through a T-mesh  $\mathcal{T}$  (see Figure 2) and a valid knot interval configuration, both of which are defined through CAD software. In contrast to NURBS discretisations which consist of a patchwork of structured grids, T-splines allow for local refinement and guarantee water-tight models. In the present study we adopt analysis-suitable T-splines [32] which satisfy important analysis properties including linear-independence and partition of unity. In the interests of brevity, we do not wish to delve into the technical details of how to construct T-spline basis functions and instead we give a brief overview.

We assume that the boundary of the domain  $\Gamma \in \mathbb{R}^3$  is defined by a the geometric mapping provided by an analysis-suitable T-mesh  $\mathcal{T}$  with 4-dimensional control points  $\{\mathbf{P}_A\}_{A=1}^{n_{cp}}$ ,  $\mathbf{P}_A = (x_A, y_A, z_A, w_A) = (x_A^1, x_A^2, x_A^3, w_A)$  where  $w_A$  denotes a control point weighting. Bézier extraction [43] is performed such that the T-mesh is reduced to a set of Bézier elements  $\{\Gamma^e\}_{e=1}^{n_e}$  with extraction operators  $\mathbf{C}^e$  that form a decomposition of the boundary  $\Gamma = \cup_{e=1}^{n_e} \overline{\Gamma^e}$  with  $\Gamma_i \cap \Gamma_j = \emptyset, i \neq j$ . The motivation for Bézier extraction is to reduce the T-spline discretisation to a set of elements with a common structure (i.e. equal number of non-zero basis functions) that facilitates implementation in analysis codes and allows for faster computations.

We let  $\tilde{\Gamma} = [-1, 1]^2$  denote the local parent domain with coordinate  $\tilde{\xi} \in \tilde{\Gamma}$ . Local to global index mappings  $A = \text{IEN}(a, e)$  and  $A = \widehat{\text{IEN}}(a, e)$  are prescribed such that a continuous and semi-discontinuous basis can be written as

$$R_A(\mathbf{x}(\tilde{\xi}))|_e = R_{\text{IEN}(a,e)}(\mathbf{x}(\tilde{\xi}))|_e = R_a^e(\tilde{\xi}) \quad (10)$$

and

$$\hat{R}_A(\mathbf{x}(\tilde{\xi}))|_e = \hat{R}_{\widehat{\text{IEN}}(a,e)}(\mathbf{x}(\tilde{\xi}))|_e = \hat{R}_a^e(\tilde{\xi}). \quad (11)$$

Further details on the construction of the continuous and semi-discontinuous T-spline basis can be found in [45]. In matrix form, the local rational T-spline basis functions are computed as

$$\mathbf{R}^e(\tilde{\xi}) = \mathbf{C}^e \mathbf{N}^e(\tilde{\xi}) \quad (12)$$

where  $\mathbf{N}^e$  is a matrix of Bernstein polynomial functions. This expression is applied to both (10) and (11) with the semi-discontinuous basis accounted for through the mapping  $\widehat{\text{IEN}}(a, e)$ . The element T-spline geometric map  $\mathbf{x} : \tilde{\Gamma} \rightarrow \Gamma^e$  is defined as

$$\mathbf{x}(\tilde{\xi}) = \sum_{a=1}^n \mathbf{P}_a^e R_a^e(\tilde{\xi}) \quad (13)$$

where  $\mathbf{P}_{\text{IEN}(a,e)} = \mathbf{P}_a^e$  and  $n$  denotes the number of local non-zero basis functions defined over element  $e$ .

In the present study we adopt T-splines to discretise the geometry through expression (13) and construct continuous and semi-discontinuous bases by substituting expressions (10) and (11) into (5) giving

$$u_D(\mathbf{y}) = \sum_{J=1}^{n_d} \phi_J^D R_J(\mathbf{y}) \quad u_N(\mathbf{y}) = \sum_{J=1}^{n_n} \phi_J^N \hat{R}_J(\mathbf{y}). \quad (14)$$

#### 4. Fast multipole isogeometric boundary element method

Our attention now turns to the algorithm used to accelerate far-field computations for the operators given in (3). Our task is to express such operators in a form amenable for computation through the black-box FMM. Care must be taken however in the computation of singular boundary integrals that precludes the straightforward use of the black-box FMM. We also note that in the context of the boundary element method the

FMM is used to compute the matrix-vector operator given by the left hand side of (9) demanding an iterative solver approach. We therefore rewrite (9) as

$$\begin{aligned} [K]_{IJ} \{\phi\}_J &= \{f\}_I \\ \{L[\phi]\}_I &= \{f\}_I \end{aligned} \quad (15)$$

which is now in a form amenable for an iterative solver such as GMRES [40]. We now specify the construction of the operator  $\{L[\phi]\}_I$  and force vector  $\{f\}_I$  for accelerated computations.

#### 4.1. Integral operators

We first define two boundary integral operators corresponding to unknown Dirichlet and Neumann data respectively as

$$L_D[\phi](\mathbf{x}) = \int_{\Gamma_N} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} \phi(\mathbf{y}) d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma \quad (16)$$

$$L_N[\phi](\mathbf{x}) = \int_{\Gamma_D} G(\mathbf{x}, \mathbf{y}) \frac{\partial \phi(\mathbf{y})}{\partial n} d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma \quad (17)$$

with

$$L[\phi](\mathbf{x}) = L_D[\phi](\mathbf{x}) + L_N[\phi](\mathbf{x}). \quad (18)$$

In future notation we drop the term  $[\phi]$  in each of these operators for succinctness. Likewise, we define operators corresponding to known Dirichlet and Neumann data respectively as

$$f_D(\mathbf{x}) = \int_{\Gamma_D} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} g_D(\mathbf{y}) d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma \quad (19)$$

$$f_N(\mathbf{x}) = \int_{\Gamma_N} G(\mathbf{x}, \mathbf{y}) g_N(\mathbf{y}) d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma \quad (20)$$

with

$$f(\mathbf{x}) = f_D(\mathbf{x}) + f_N(\mathbf{x}). \quad (21)$$

Each operator defined in (16), (17), (19) and (20) can be decomposed into a singular and far-field component which, in the case of (16) is defined as

$$L_D(\mathbf{x}) = L_D^s(\mathbf{x}) + L_D^{far}(\mathbf{x}) \quad (22)$$



with the singular component defined by

$$L_D^s(\mathbf{x}) = \int_{\Gamma_N \cap \Gamma_s} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} \phi(\mathbf{y}) d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma \quad (23)$$

and the far-field component written as

$$L_D^{far}(\mathbf{x}) = \int_{\Gamma_N \setminus \Gamma_s} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} \phi(\mathbf{y}) d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma. \quad (24)$$

We defer a formal definition of the singular region of the boundary  $\Gamma_s$  until Section 4.3. Equivalent expressions for the remaining operators (17), (19), (20) are given in Appendix A.

#### 4.2. Recasting integral operators into summations of point charge interactions

We now see the general approach to evaluate the components of (15). Letting  $(\bullet)(\mathbf{x})$  denote the operator given by either (16), (17), (19) or (20) we write its decomposition as

$$(\bullet)(\mathbf{x}) = (\bullet)^s(\mathbf{x}) + (\bullet)^{far}(\mathbf{x}) \quad (25)$$

where  $(\bullet)^s(\mathbf{x})$  and  $(\bullet)^{far}(\mathbf{x})$  are sums of singular terms and far-field terms respectively. Applying a collocation approach this is discretised as

$$(\bullet)(\mathbf{x}_I) = (\bullet)^s(\mathbf{x}_I) + (\bullet)^{far}(\mathbf{x}_I) \quad I = 1, 2, \dots, n_c \quad (26)$$

where standard singular quadrature methods are applied to compute  $(\bullet)^s(\mathbf{x}_I)$  and the black-box FMM is applied to compute  $(\bullet)^{far}(\mathbf{x}_I)$  by recasting it into the form given by (1).

The first step is to construct the data structures which are required by the black-box FMM. This amounts to sets of source points and charges corresponding to each of the operators defined by (16), (17), (19) and (20). A set of source points  $\{\mathbf{y}_J\}_{J=1}^{N_s}$  is defined through a quadrature rule  $\{\tilde{\boldsymbol{\xi}}_j, w_j\}_{j=1}^{n_{gp}}$  as

$$\mathbf{y}_J = \mathbf{y}_{J(e,j)} = \sum_{a=1}^n \mathbf{P}_a^e R_a^e(\tilde{\boldsymbol{\xi}}_j) \quad e = 1, 2, \dots, n_{el} \quad j = 1, 2, \dots, n_{gp} \quad (27)$$

where  $J(e, j)$  is a mapping from an element and gauss point index to a global source point index. Similarly, a set of weights  $\{w_J\}_{J=1}^{N_s}$  comprised of Jacobian determinants

and quadrature weights is defined through

$$w_J = w_{J(e,j)} = |J^e(\tilde{\xi}_j)|w_j \quad e = 1, 2, \dots, n_{el} \quad j = 1, 2, \dots, n_{gp}. \quad (28)$$

To enable the use of the black-box fast multipole algorithm, kernels for double-layer and single-layer potentials are defined as

$$\mathbf{K}^{DL}(\mathbf{x}, \mathbf{y}) = \frac{\mathbf{y} - \mathbf{x}}{|\mathbf{x} - \mathbf{y}|^3} \quad K^{SL}(\mathbf{x}, \mathbf{y}) = \frac{1}{|\mathbf{x} - \mathbf{y}|} \quad (29)$$

with corresponding charge operators

$$\boldsymbol{\sigma}^{DL}[\mathbf{y}, f(\mathbf{y})] = -\frac{1}{4\pi} \mathbf{n}(\mathbf{y}) f(\mathbf{y}) \quad \sigma^{SL}[\mathbf{y}, f(\mathbf{y})] = \frac{1}{4\pi} f(\mathbf{y}) \quad (30)$$

where  $\mathbf{n}(\mathbf{y})$  is the outward pointing normal and  $f(\mathbf{y})$  represents a given boundary function. Using (30) we define the following sets of field-point and charge pairs

$$Q^{K,D} = \{ (\mathbf{y}_J, \boldsymbol{\sigma}_J) : \mathbf{y}_J \in \Gamma_N \setminus \Gamma_s, \boldsymbol{\sigma}_J = w_J \boldsymbol{\sigma}^{DL}[\mathbf{y}_J, \phi(\mathbf{y}_J)] \} \quad (31)$$

$$Q^{K,N} = \{ (\mathbf{y}_J, \sigma_J) : \mathbf{y}_J \in \Gamma_D \setminus \Gamma_s, \sigma_J = w_J \sigma^{SL}[\mathbf{y}_J, \partial\phi(\mathbf{y}_J)/\partial n] \} \quad (32)$$

$$Q^{f,D} = \{ (\mathbf{y}_J, \boldsymbol{\sigma}_J) : \mathbf{y}_J \in \Gamma_D \setminus \Gamma_s, \boldsymbol{\sigma}_J = w_J \boldsymbol{\sigma}^{DL}[\mathbf{y}_J, g_D(\mathbf{y})] \} \quad (33)$$

$$Q^{f,N} = \{ (\mathbf{y}_J, \sigma_J) : \mathbf{y}_J \in \Gamma_N \setminus \Gamma_s, \sigma_J = w_J \sigma^{SL}[\mathbf{y}_J, g_N(\mathbf{y})] \} \quad (34)$$

which allow the required far-field expressions to be written as

$$L_D^{far}(\mathbf{x}_I) \approx \sum_{(\mathbf{y}_J, \boldsymbol{\sigma}_J) \in Q^{K,D}} \mathbf{K}^{DL}(\mathbf{x}_I, \mathbf{y}_J) \boldsymbol{\sigma}_J \quad (35)$$

$$L_N^{far}(\mathbf{x}_I) \approx \sum_{(\mathbf{y}_J, \sigma_J) \in Q^{K,N}} K^{SL}(\mathbf{x}_I, \mathbf{y}_J) \sigma_J \quad (36)$$

$$f_D^{far}(\mathbf{x}_I) \approx \sum_{(\mathbf{y}_J, \boldsymbol{\sigma}_J) \in Q^{f,D}} \mathbf{K}^{DL}(\mathbf{x}_I, \mathbf{y}_J) \boldsymbol{\sigma}_J \quad (37)$$

$$f_N^{far}(\mathbf{x}_I) \approx \sum_{(\mathbf{y}_J, \sigma_J) \in Q^{f,N}} K^{SL}(\mathbf{x}_I, \mathbf{y}_J) \sigma_J \quad (38)$$

$$I = 1, 2, \dots, N_f.$$

A schematic illustration of the sets  $Q^{K,N}$  and  $Q^{K,D}$  is given in Figure 3. We note that the use expressions (35) to (38) retain the benefits of an isogeometric formulation of exact CAD geometry and high-order (T-spline) basis functions.

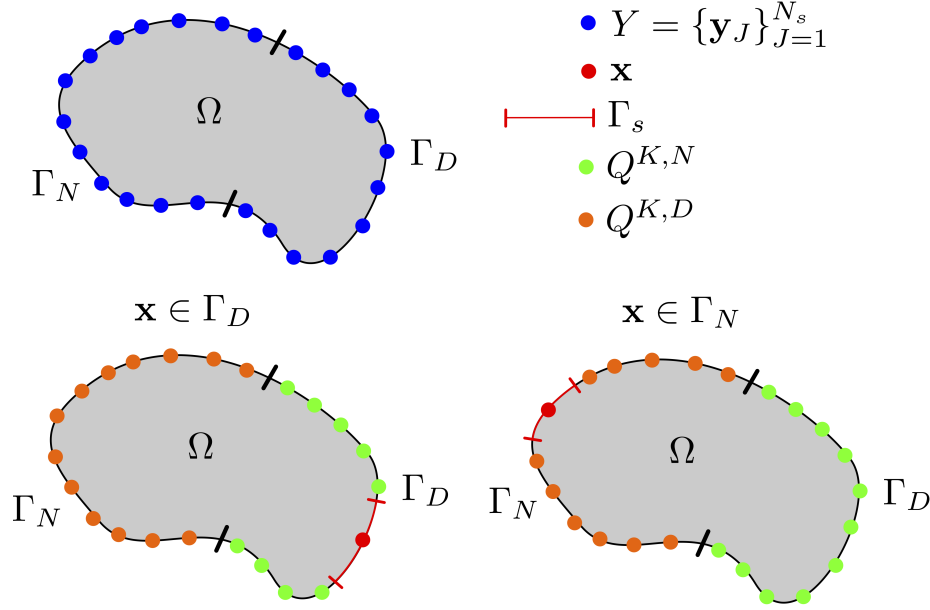


Figure 3: Illustration of field point sets used for far field computations with the black-box fast multipole approach.

The sets defined by (31) to (34) are found to depend on the field point  $\mathbf{x}_I$  due to the definition of the boundary  $\Gamma_s$ . For ease of implementation, it is often simpler to apply expressions (35) to (38) to the boundaries  $\Gamma_N$  and  $\Gamma_D$  and thereafter subtract terms related to  $\Gamma_s$ . This approach is adopted in Sec. 4.4.

Remaining singular terms are calculated through an appropriate singular quadrature scheme such as the polar transformation technique detailed in [45].

#### 4.3. Decomposition of space: octree initialisation

The final task is to outline the parameters that define an octree subdivision of space and allow definitions of singular and far-field domains to be made. A bounding box of the domain is constructed through the strong convex hull property of T-spline surfaces by first defining

$$x_{min}^i := \min_A(x_A^i) \quad x_{max}^i := \max_A(x_A^i) \quad (39)$$

and

$$\Delta^i = |x_{max}^i - x_{min}^i| \quad s^i = \frac{1}{2}(x_{max}^i + x_{min}^i) \quad (40)$$

allowing an approximate bounding box to be expressed as  $\bigotimes_{i=1}^3 [x_{min}^i, x_{max}^i]$  with maximum edge length  $L = \max(\Delta^i)$  and centre  $\mathbf{s} = (s^1, s^2, s^3)$ . We define the cube bounding box as  $\Omega_D = \bigotimes_{i=1}^3 [s^i - L/2, s^i + L/2]$ .

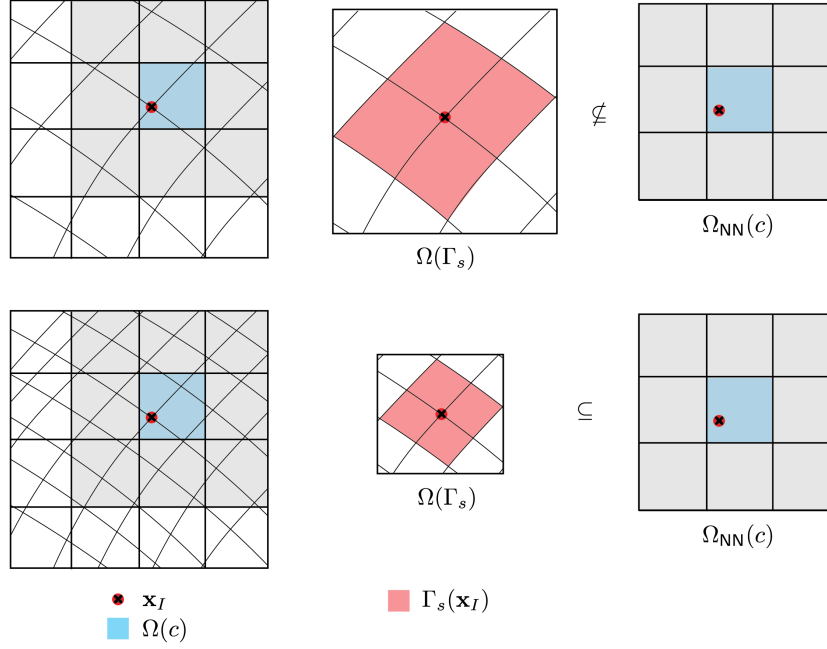


Figure 4: Illustration of the bounding boxes  $\Omega(c)$ ,  $\Omega(\Gamma_s)$  and  $\Omega_{NN}(c)$  including the subset of the boundary  $\Gamma_s$ . Thin lines in the left-hand figures represent element boundaries. Two scenarios are illustrated: the top case fails the criteria that the bounding box of  $\Gamma_s$  is a subset of  $\Omega_{NN}(c)$  indicating a coarser level of octree subdivision should be used; the bottom case fulfills the criteria and the octree can be used for BE analysis.

We determine the number of octree subdivisions required by keeping in mind quadrature routines for singular integrals. Adopting the notation of Section 2, an octree with  $m + 1$  levels and uniform refinement is assumed with the set of cells in each level defined by  $\mathcal{Y}^k$ ,  $k = 0, 1, 2, \dots, m$ . A cell at level  $k$  is denoted by  $c \in \mathcal{Y}^k$  with domain  $\Omega(c)$ . The near neighbour list of  $c$  is written as  $NN(c)$  with domain  $\Omega_{NN}(c) = \bigcup_a \overline{\Omega}(c_a) \forall c_a \in NN(c)$ .

The set of element indices containing a collocation point  $\mathbf{x}_I$  is written as  $E(\mathbf{x}_I) =$

$\{e : \mathbf{x}_I \in \Gamma_e\}$  which define a subset of the boundary  $\Gamma_s(\mathbf{x}_I) = \bigcup_{e \in \mathbf{E}(\mathbf{x}_I)} \Gamma_e$ . In future, we commonly drop the function argument  $\mathbf{x}_I$  from  $\Gamma_s$  and  $\mathbf{E}$  where it is implied by its context. Defining the bounding box of  $\Gamma_s$  as  $\Omega(\Gamma_s)$ , the criteria for terminating subdivision can now be stated as: working at the lowest octree level  $m$ , for every  $\mathbf{x}_I \in \Omega(c)$  where  $c \in \mathcal{Y}^m$ , it must be true that  $\Omega(\Gamma_s) \subseteq \Omega_{\text{NN}}(c)$ . That is, the bounding box of all boundary elements containing the point  $\mathbf{x}_I$  must be a subset of the domain defined by the nearest neighbours of the cell that contains  $\mathbf{x}_I$ . This is illustrated graphically in Figure 4. Assuming uniform octree subdivision, this is implemented practically as

$$m = \left\lceil \frac{\ln(\sqrt{3}/h_{\max})}{\ln 2} \right\rceil \quad (41)$$

with the normalised maximum element length  $h_{\max}$  given by

$$h_{\max} = \max_e (\text{diam}(\Gamma_e)) / L. \quad (42)$$

#### 4.4. Algorithm

The basic algorithm for the present fast multipole implementation is outlined in Figure 5. The specific details of how the operator  $L(\mathbf{x})$  is constructed through the black-box FMM are detailed in Algorithm 1 with the associated correction of singular terms detailed in Algorithm 2. The functions `evalDLSingular()` and `evalSLSingular()` in Algorithm 2 refer to singular quadrature routines that compute integrals of the double-layer and single-layer potential respectively. Standard transformation techniques such as the Duffy transformation [16] or a polar-coordinate transformation [45] can be used to implement these functions.

The black-box FMM code used for far-field computations in the present study can be found at <https://bitbucket.org/rns/bbfmm3d/> which includes examples of its usage.

## 5. Numerical examples

### 5.1. Torus example

To illustrate the asymptotic behaviour of the present black-box fast multipole implementation we solve Laplace's equation imposed over a torus geometry with outer

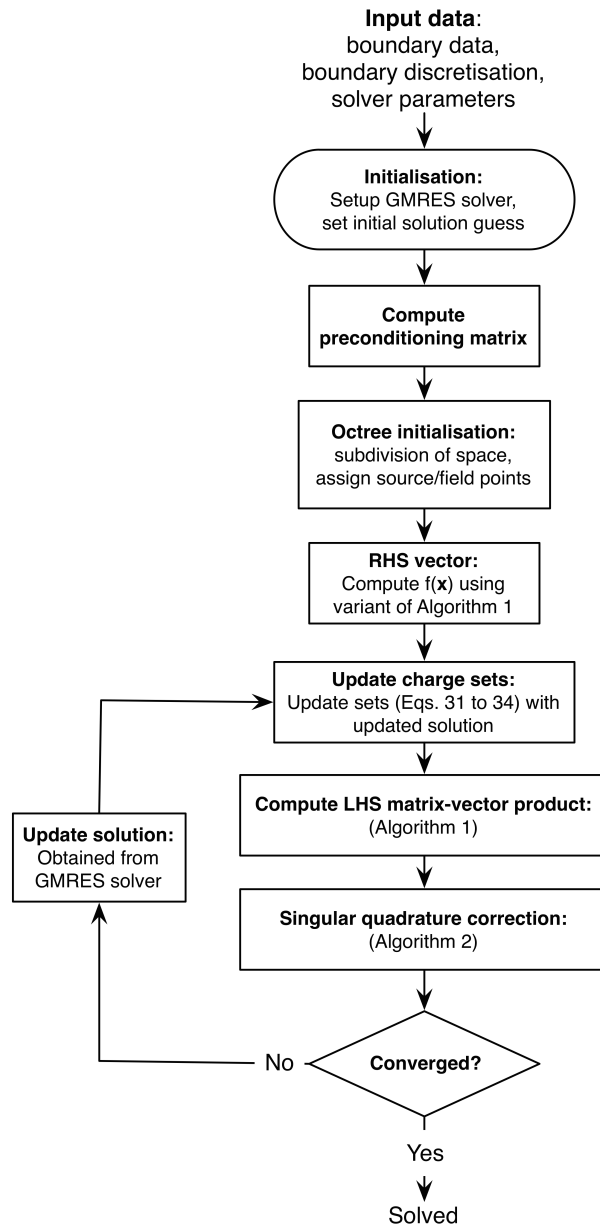


Figure 5: Basic algorithm of the present black-box fast multipole solver.

---

**Algorithm 1:** Fast multipole approximation of operator  $L(\mathbf{x}_I)$  from updated solution  $\{\phi\}$ .

---

**Input** :  $\Gamma_N, \Gamma_D, \phi = [\phi_D \mid \phi_N]^T, \{\mathbf{x}_I\}_{I=1}^{N_f}, \{\mathbf{P}_A\}_{A=1}^{n_{cp}}, \{R_J(\mathbf{y})\}_{J=1}^{n_d}, \{\hat{R}_J(\mathbf{y})\}_{J=1}^{n_n}, \{\Gamma_e\}_{e=1}^{n_{el}}, \{(\tilde{\xi}_j, w_j)\}_{j=1}^{n_{gp}}$

**Output:** FMM approximation of  $L(\mathbf{x}_I)$

$e \leftarrow 0;$

**while**  $e < n_{el}$  **do**

$j \leftarrow 0;$

**while**  $j < n_{gp}$  **do**

$\mathbf{y}_{J(e,j)} \leftarrow \sum_{a=1}^n \mathbf{P}_a^e R_a^e(\tilde{\xi}_j);$

$w_{J(e,j)} \leftarrow |J^e(\tilde{\xi}_j)|w_j;$

**if**  $\Gamma^e \in \Gamma^N$  **then**

            // Compute double-layer points and charges

$\phi(\mathbf{y}_J) \leftarrow \sum_{a=1}^n \phi_D^{a,e} R_a^e(\tilde{\xi}_j);$

$\sigma_J \leftarrow w_J \sigma^{DL}[\mathbf{y}_J, \phi(\mathbf{y}_J)];$

$Q^{K,D}.\text{insert}(\mathbf{y}_J, \sigma_J);$

**else**

            // Compute single-layer points and charges

$\partial\phi(\mathbf{y}_J)/\partial n \leftarrow \sum_{a=1}^n \phi_N^{a,e} \hat{R}_a^e(\tilde{\xi}_j);$

$\sigma_J \leftarrow w_J \sigma^{SL}[\mathbf{y}_J, \partial\phi(\mathbf{y}_J)/\partial n];$

$Q^{K,N}.\text{insert}(\mathbf{y}_J, \sigma_J);$

**end**

$j \leftarrow j + 1$

**end**

$e \leftarrow e + 1$

**end**

// Black-box algorithm of [18]

$L_D(\mathbf{x}_I) \leftarrow \sum_{(\mathbf{y}_J, \sigma_J) \in Q^{K,D}} \mathbf{K}^{DL}(\mathbf{x}_I, \mathbf{y}_J) \sigma_J;$

$L_N(\mathbf{x}_I) \leftarrow \sum_{(\mathbf{y}_J, \sigma_J) \in Q^{K,N}} K^{SL}(\mathbf{x}_I, \mathbf{y}_J) \sigma_J;$

$L(\mathbf{x}_I) \leftarrow L_D(\mathbf{x}_I) + L_N(\mathbf{x}_I)$

---

---

**Algorithm 2:** Correction of singular terms for operator  $L(\mathbf{x}_I)$ 


---

**Input** :  $Q^{K,D}, Q^{K,N}, \phi = [\phi_D \mid \phi_N]^T, \{\mathbf{x}_I\}_{I=1}^{N_f}, L(\mathbf{x}_I), \{\Gamma_e\}_{e=1}^{n_{el}}$

**Output:**  $L(\mathbf{x}_I)$

$I \leftarrow 0;$

**while**  $I < N_f$  **do**

**foreach**  $\Gamma_e \in \Gamma_s(\mathbf{x}_I)$  **do**

**if**  $\Gamma_e \in \Gamma_N$  **then**

$$L(\mathbf{x}_I) \leftarrow L(\mathbf{x}_I) - \sum_{\substack{(\mathbf{y}_J, \sigma_J) \in Q^{K,D} \\ \mathbf{y}_J \in \Gamma_s}} \mathbf{K}^{DL}(\mathbf{x}_I, \mathbf{y}_J) \sigma_J ;$$

$$L(\mathbf{x}_I) \leftarrow L(\mathbf{x}_I) + \text{evalDLSingular}(\mathbf{x}_I, \Gamma_e, \{\phi_D^{a,e}\}_{a=1}^n);$$

**else**

$$L(\mathbf{x}_I) \leftarrow L(\mathbf{x}_I) - \sum_{\substack{(\mathbf{y}_J, \sigma_J) \in Q^{K,N} \\ \mathbf{y}_J \in \Gamma_s}} K^{SL}(\mathbf{x}_I, \mathbf{y}_J) \sigma_J ;$$

$$L(\mathbf{x}_I) \leftarrow L(\mathbf{x}_I) + \text{evalSLSingular}(\mathbf{x}_I, \Gamma_e, \{\phi_N^{a,e}\}_{a=1}^n);$$

**end**

**end**

$I \leftarrow I + 1;$

**end**

---



radius  $r_o = 2$  and inner radius  $r_i = 1$ . We evaluate both solution runtimes and memory consumption through successive levels of uniform h-refinement (knot insertion) making use of bivariate T-spline basis functions of degree  $\mathbf{p} = (p_1, p_2) = (3, 3)$  to discretise both the geometry and boundary fields during analysis. We note that the T-spline torus geometry discretisation is equivalent to a NURBS discretisation. Singular integrals are evaluated through a polar integral transformation and a regularisation procedure as outlined in [45]. A  $(p_1 + 1) \times (p_2 + 1)$  Gauss-Legendre quadrature rule is used to evaluate all numerical integrals. Dirichlet boundary conditions of  $\phi(\mathbf{x}) = x$  are imposed over the entire boundary by performing an  $L_2$  projection onto the T-spline basis. The solution to this problem is given by  $q = \partial\phi(\mathbf{x})/\partial n = n_x$ .

We compare the accelerated black-box approach against a direct solver approach in which an LU solver is employed [27]. Both solvers make use of the same quadrature rule as noted above with far-field terms in the black-box FMM computed with a tolerance of  $\varepsilon = 10^{-5}$  and Chebyshev interpolation with  $n_{ch} = 5$ . We choose these parameters since they strike a compromise between accuracy and solution runtimes as illustrated in the parameter study in Appendix B. A GMRES iterative solver was adopted for the black-box approach with a solver tolerance of  $10^{-5}$  prescribed. We note that no adaptive quadrature is used in the direct solver case. All simulations were performed on a 2.4GHz quadcore processor allowing for a maximum of 8 parallel threads due to hyperthreading. In both solvers the assembly process was parallelised by dividing collocation points into an appropriate number of work units.

Figures 6a and 6b illustrate the coarsest and finest meshes respectively with the generated octree and boundary solution for the finest discretisation shown in Figure 7. Results of runtime, relative  $L_2$  error, maximum pointwise error and memory usage are tabulated in Table 1. To illustrate asymptotic behaviour, solution runtimes for both solvers are plotted in Figure 8 in which the  $O(N)$  behaviour of the black-box solver is demonstrated and runtimes are consistently lower than the LU solver for all meshes. An  $O(N^2)$  asymptotic behaviour is observed for the LU solver indicating matrix assembly dominates solution runtime. For higher degrees of freedom this is expected to behave as  $O(N^3)$  when LU factorisation dominates runtime. Additionally, representative times for far-field and near-field computations are given for each of the torus

mesh	dof	runtime (s)		memory (MB)		$\frac{\ q-q^h\ _{L_2}}{\ q\ _{L_2}}$		$\frac{\ q-q^h\ _{\infty}}{\ q\ _{\infty}}$	
		FMM	LU	FMM	LU	FMM	LU	FMM	LU
0	144	9	10	130	150	2.356E-3	7.909E-2	5.890E-3	1.349E-1
1	256	19	25	213	256	1.429E-3	4.637E-2	3.709E-3	1.024E-1
2	576	42	125	442	639	1.785E-3	2.552E-2	5.711E-3	8.999E-2
3	1,600	126	1,061	1,163	1,835	1.492E-3	1.492E-2	1.137E-2	7.929E-2
4	5,184	420	13,485	3,604	6,308	2.087E-3	9.528E-3	1.943E-2	8.833E-2
5	18,496	1,638	180,533	11,158	23,900	2.928E-3	6.599E-3	4.728E-2	9.102E-2

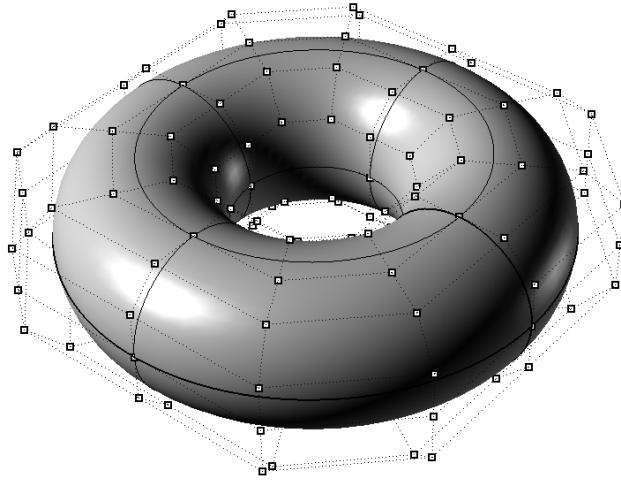
Table 1: T-spline torus problem with cubic basis: solver runtime, memory usage and relative errors.

discretisations in Table 2 where it is observed that near-field computations dominate in all cases, becoming less dominant for finer meshes. Speeding up near-field computations is therefore a prime candidate for future speed improvements to the present approach.

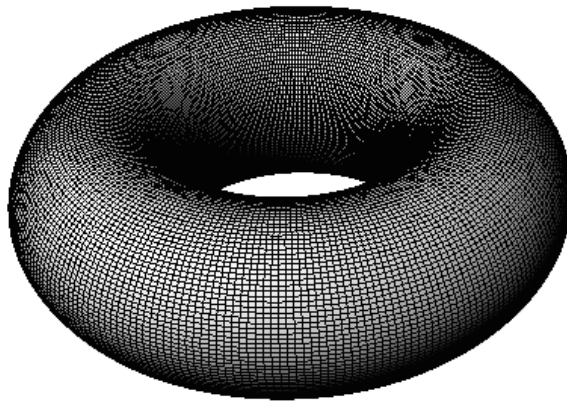
mesh	dof	typical farfield	typical nearfield	nearfield/ farfield
		(s)	(s)	
0	144	4.989E-3	1.687E-1	33.8
1	256	2.429E-2	3.294E-1	13.6
2	576	9.587E-2	6.037E-1	6.3
3	1,600	3.821E-1	1.570	4.1
4	5,184	1.462	4.866	3.3
5	18,496	7.176	18.110	2.5

Table 2: Typical farfield and nearfield computation times corresponding to each torus discretisation.

Figure 9 plots the memory consumption for each solver where it can be seen that the direct solver consistently requires more memory, albeit relatively similar in magnitude to the black-box solver. Both solvers exhibit  $O(N)$  behaviour but it is expected that for very large degrees of freedom, the  $O(N^2)$  storage requirements of the direct



(a) Mesh 0: 144 control points and 16 T-spline elements.



(b) Mesh 5: 18,496 control points and 16,384 T-spline elements (control points omitted for clarity).

Figure 6: Torus T-spline geometry discretisations.

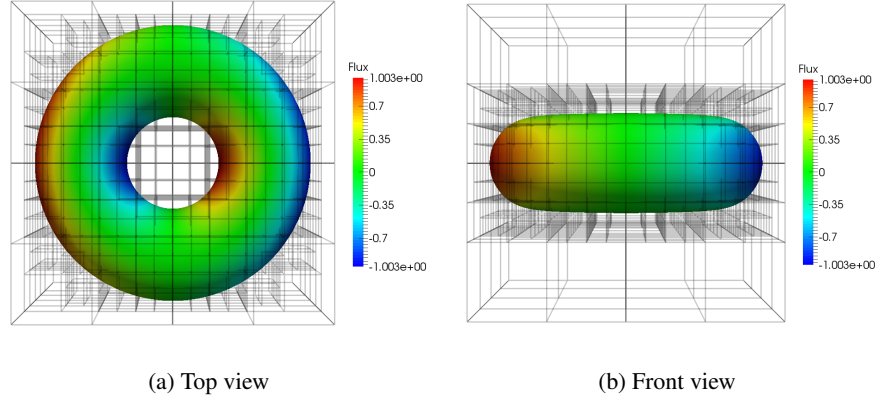


Figure 7: Octree subdivision for mesh 5 with numerical flux solution for black-box solver.

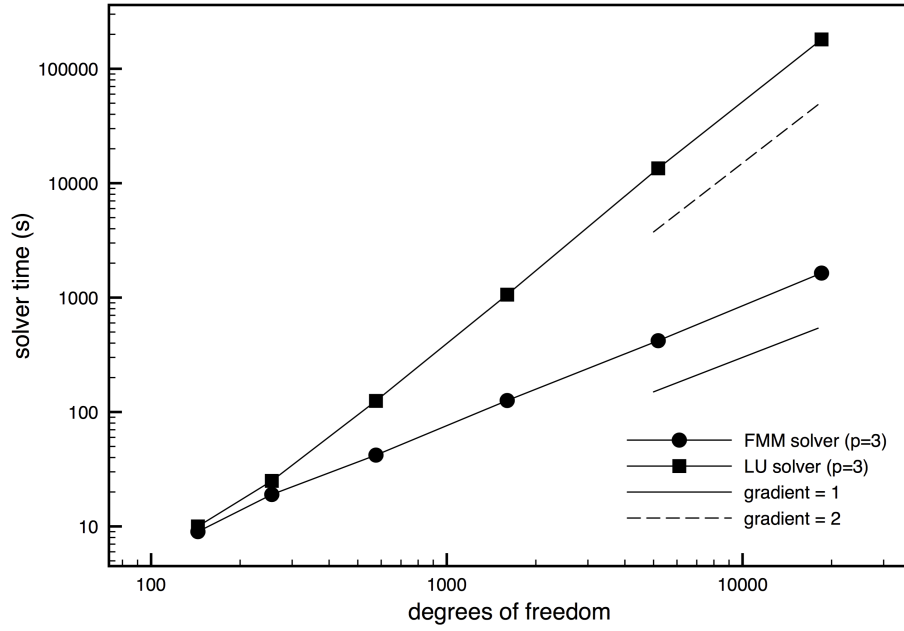


Figure 8: Comparison of runtimes for the accelerated black-box approach and a direct solver applied to a Laplace problem posed over cubic T-spline torus discretisations.

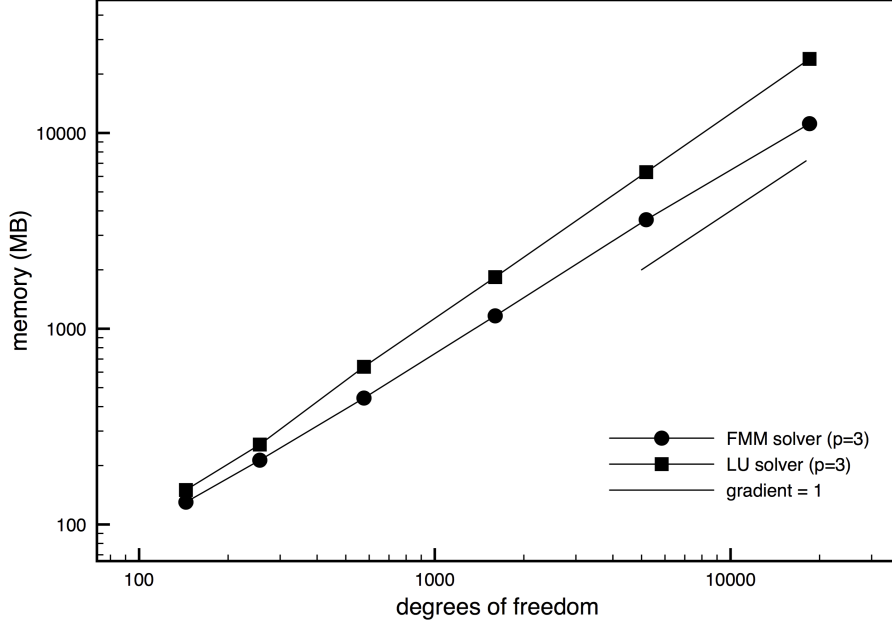


Figure 9: Comparison of memory usage for the accelerated black-box approach and a direct solver applied to a Laplace problem posed over cubic T-spline torus discretisation.

solver will dominate. In all the examples considered in the present study the dominant source of memory usage is the use of cache algorithms that store data such as Bernstein basis function values, normal components and physical coordinates corresponding to quadrature points. Such algorithms provide substantial speed benefits, but further work on how to efficiently store such data while minimising memory usage is the subject of future research.

Inspection of the relative  $L_2$  error and maximum pointwise error for each torus discretisation in Table 1 reveals that *the black-box FMM approximation is capable of delivering higher accuracies in faster runtimes* compared to standard quadrature schemes with a direct solver. It is noted however, that a slight increase in relative errors is seen for the finest discretisations (meshes 4 and 5). This can be explained by inspecting Figure 10 which illustrates the pointwise error for the coarsest and finest mesh and clearly demonstrates that regions of maximum pointwise error are located at points of  $C^0$  con-

<b>model</b>	<b>dof</b>	<b>T-spline elements</b>	<b>Bézier elements</b>	<b>runtime (s)</b>	<b>memory (MB)</b>	$\frac{\ q-q^h\ _{L_2}}{\ q\ _{L_2}}$	$\frac{\ q-q^h\ _{\infty}}{\ q\ _{\infty}}$
Probe	8,126	6,576	11,034	2,317	10,846	2.993E-03	1.960E-01
Molecule	14,820	14,880	51,600	3,782	15,206	5.266E-03	4.552E-02

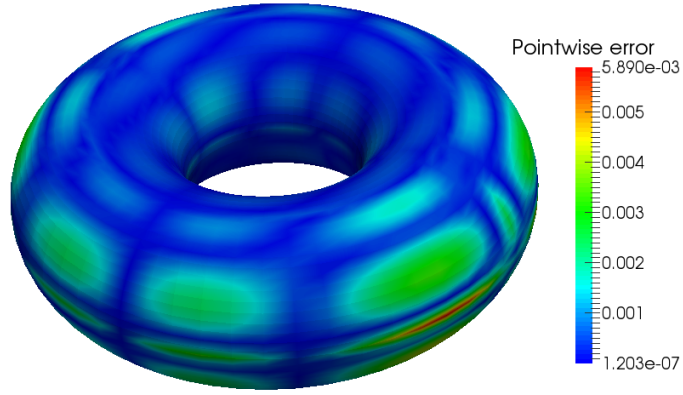
Table 3: Summary of discretisations, solver performance and error magnitudes for probe and molecule models.

tinuity (patch boundaries) attributable to quadrature error in nearly singular integrals. Further comments on quadrature error will be made in Section 5.2.

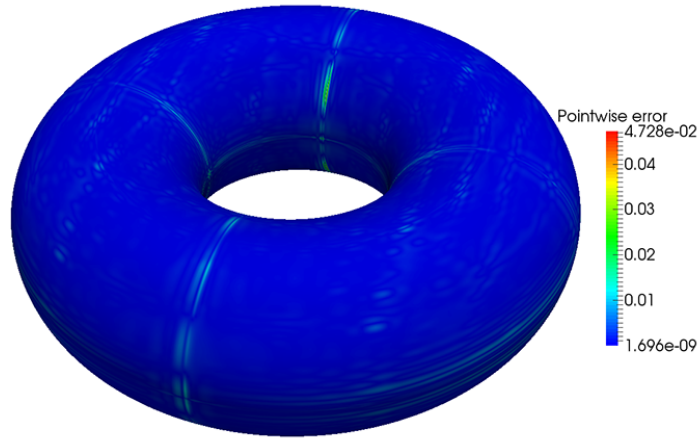
### 5.2. Integration of CAD and accelerated BE analysis

One of the fundamental goals of the present study is to demonstrate acceleration algorithms for analysis of models generated directly from CAD software. We demonstrate this through two T-spline models generated in Rhino® as illustrated in Figures 11a and 11b, both discretised using cubic T-splines and which cannot be solved using a direct LU solver with the present hardware. The model in Figure 11a represents a probe that is used for measuring acoustic pressure discretised with 6,576 T-spline elements (faces) and 8,126 control points. Closeup images of the probe are shown in Figures 12a and 12b which illustrate T-spline elements and Bézier elements respectively. The model exhibits  $C^0$  surfaces that lead to a discontinuous flux solution that necessitates a discontinuous discretisation. Identical boundary conditions as prescribed for the torus problem in Section 5.1 were prescribed. As detailed in Table 3, the black-box FMM solver generated a solution in 2,317s (38mins 37s) with a relative  $L_2$  error of  $2.993 \times 10^{-3}$ . The numerical solution and octree discretisation for this particular discretisation is illustrated in Figures 13a and 13b.

The second example considered is shown in Figure 11b which is intended to illustrate the use of the present approach for molecular electrostatic computations that are commonplace in computational chemistry (e.g. [54]). As before, the model was generated in Rhino® using T-splines and decomposed into a set of 51,600 Bézier elements using the IGA plugin of [44]. A closeup of the control grid is shown in Figure 14a with the associated Bézier mesh shown in Figure 14b. To mimic boundary

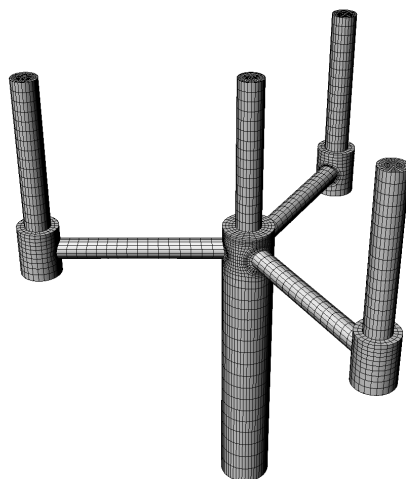


(a) Mesh 0

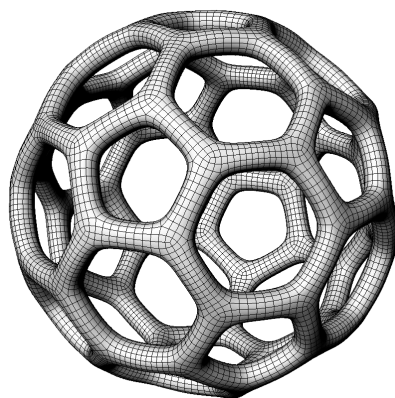


(b) Mesh 5

Figure 10: Torus study: pointwise error  $\frac{\|q - q^h\|_\infty}{\|q\|_\infty}$  obtained using the present black-box FMM solver.



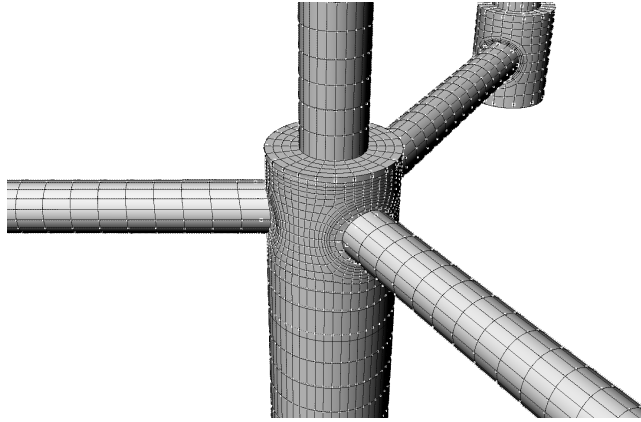
(a) T-spline probe model.



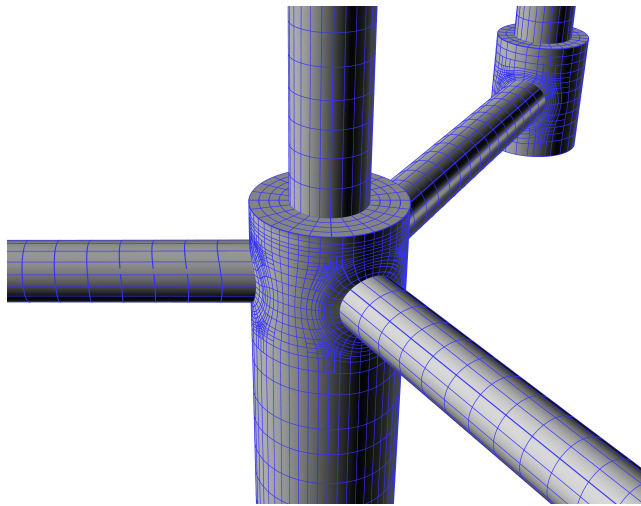
(b) T-spline molecule model.

Figure 11: T-spline models used to perform integrated design and analysis with the present black-box FMM solver.



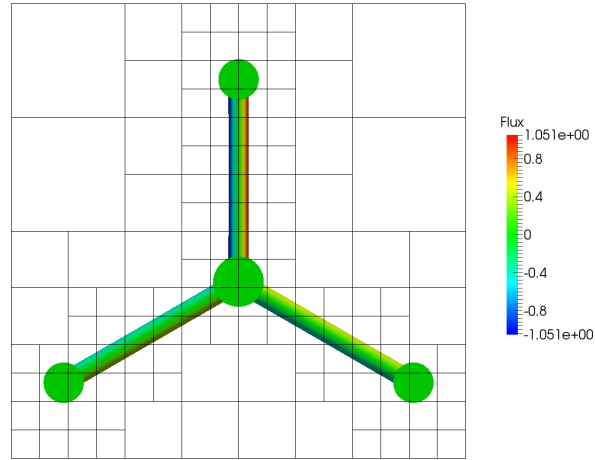


(a) T-spline elements and control points.

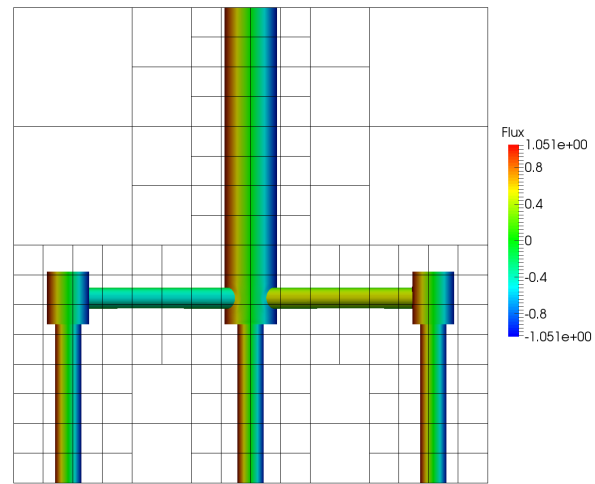


(b) Bézier elements.

Figure 12: Closeup images of the T-spline model shown in Figure 11a.



(a) Front view



(b) Top view

Figure 13: Front and top views of probe flux solution with its associated octree discretisation.

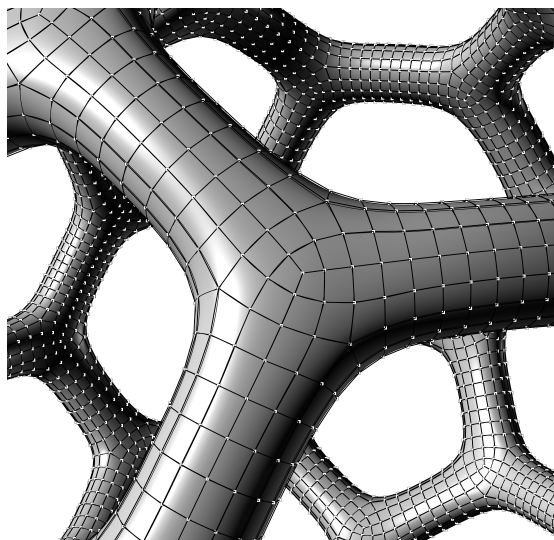
conditions commonly found in molecular electrostatic applications, Dirichlet boundary conditions were applied corresponding to a set of point charges at coordinates  $\{\mathbf{x}_i\}_{i=1}^{n_{cg}}$  as detailed in Appendix C. A boundary potential function was then specified as  $\phi(\mathbf{x}) = \sum_{i=1}^{n_{cg}} \frac{1}{|\mathbf{x} - \mathbf{x}_i|}$  applied using an  $L_2$  projection. The flux solution to this problem is given by  $\partial\phi(\mathbf{x})/\partial n = \sum_{i=1}^{n_{cg}} \nabla\phi \cdot \mathbf{n}$  with  $\nabla\phi = (-x, -y, -z)$ .

For this particular model the black-box FMM solver generated a solution in 3,782s (63mins 2s) with a relative  $L_2$  error of  $5.266 \times 10^{-3}$ . The generated flux solution is shown in Figure 15a with the associated octree discretisation used for far-field computations shown in Figure 15b. Inspection of pointwise errors as shown in Figure 16 reveals that maximum errors are concentrated around extraordinary points which is attributable to errors in the present numerical quadrature scheme. To address this, the use of a self-adaptive nested quadrature scheme which performs integration to a specified tolerance will be the subject of a future publication. But we note that even with the basic quadrature scheme adopted in the present study the maximum pointwise error is localised around extraordinary points with other regions exhibiting pointwise errors orders of magnitude less than this maximum value.

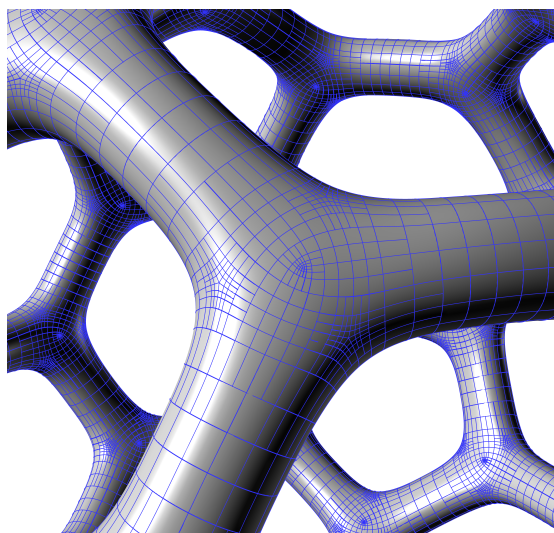
Both of these studies have demonstrated that accelerated BE analysis can be performed directly on CAD data using a black-box FMM algorithm using T-splines as a common basis for geometry and analysis. In this way we make a contribution to the ultimate goal of fully-integrated design and analysis software for efficient engineering workflows.

## 6. Conclusion

This paper outlines a method to incorporate a black-box fast multipole method within an isogeometric boundary element method for accelerated and reduced memory computations. This is achieved by decomposing boundary integral operators into singular and far-field terms in which singular terms are computed through conventional singular quadrature routines and far-field terms are approximated through the black-box fast multipole method by recasting integral operators as summations of point charge interactions. We demonstrate the behaviour of the accelerated approach for

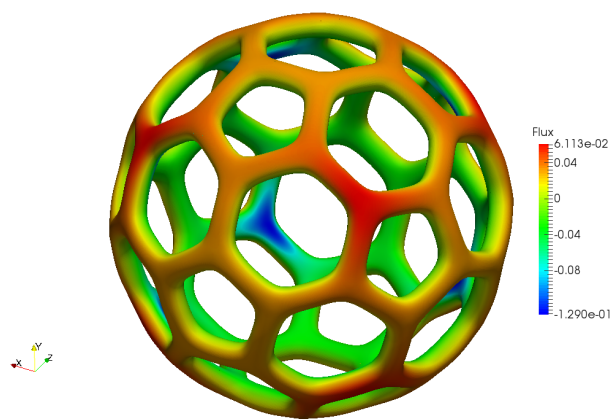


(a) Control grid

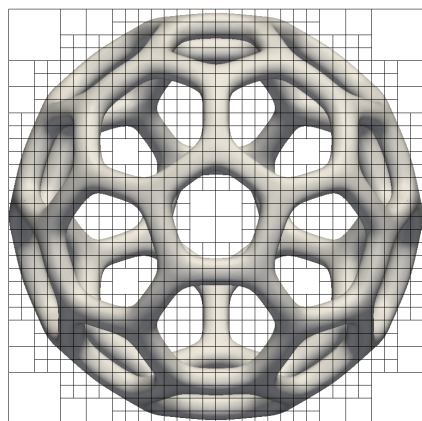


(b) Bézier elements

Figure 14: Closeup views of T-spline molecular model shown in Figure 11b

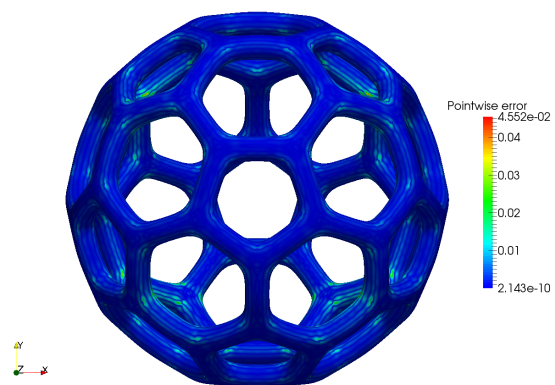


(a) Flux solution

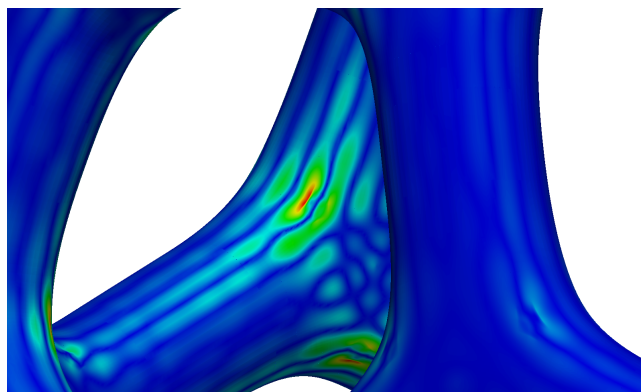


(b) Octree discretisation (front projection)

Figure 15: Molecular T-spline model flux solution and octree discretisation.



(a) Global view



(b) Closeup view around extraordinary point

Figure 16: Pointwise error of flux solution over molecule T-spline model

three-dimensional potential problems in which T-splines are used as a basis for geometry and analysis. The approach exhibits  $O(N)$  asymptotic behaviour with savings demonstrated in both solver runtime and memory consumption over a conventional direct solver. Additionally, the ability to handle geometries of arbitrary complexity is demonstrated.

The present study is focused on potential problems but the method can be easily extended to other kernels that govern applications such as elasticity, Stokes flow and medium-frequency Helmholtz problems. We believe that the black-box acceleration method introduced in this work is a suitable candidate for industrial integrated design and analysis software and provides a stepping stone towards industrial isogeometric boundary element software.

### A. Boundary integral operators

The boundary integral operators defined by (17), (19) and (20) are decomposed into singular and far field components as follows:

$$L_N(\mathbf{x}) = L_N^s(\mathbf{x}) + L_N^{far}(\mathbf{x}) \quad (43)$$

$$f_D(\mathbf{x}) = f_D^s(\mathbf{x}) + f_D^{far}(\mathbf{x}) \quad (44)$$

$$f_N(\mathbf{x}) = f_N^s(\mathbf{x}) + f_N^{far}(\mathbf{x}) \quad (45)$$

where

$$L_N^s(\mathbf{x}) = \int_{\Gamma_D \cap \Gamma_s} G(\mathbf{x}, \mathbf{y}) \frac{\partial \phi(\mathbf{y})}{\partial n} d\Gamma(\mathbf{y}) \quad (46)$$

$$L_N^{far}(\mathbf{x}) = \int_{\Gamma_D \setminus \Gamma_s} G(\mathbf{x}, \mathbf{y}) \frac{\partial \phi(\mathbf{y})}{\partial n} d\Gamma(\mathbf{y}) \quad (47)$$

$$f_D^s(\mathbf{x}) = \int_{\Gamma_D \cap \Gamma_s} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} g_D(\mathbf{y}) d\Gamma(\mathbf{y}) \quad (48)$$

$$f_D^{far}(\mathbf{x}) = \int_{\Gamma_D \setminus \Gamma_s} \frac{\partial G(\mathbf{x}, \mathbf{y})}{\partial n} g_D(\mathbf{y}) d\Gamma(\mathbf{y}) \quad (49)$$

$$f_N^s(\mathbf{x}) = \int_{\Gamma_N \cap \Gamma_s} G(\mathbf{x}, \mathbf{y}) g_N(\mathbf{y}) d\Gamma(\mathbf{y}) \quad (50)$$

$$f_N^{far}(\mathbf{x}) = \int_{\Gamma_N \setminus \Gamma_s} G(\mathbf{x}, \mathbf{y}) g_N(\mathbf{y}) d\Gamma(\mathbf{y}) \quad \forall \mathbf{x} \in \Gamma \quad (51)$$

<b>solver</b>					<b>typical</b>	<b>typical</b>		
<b>runtime</b>	$n_{ch}$	$\epsilon$	$\frac{\ q-q_h\ _{L_2}}{\ q\ _{L_2}}$	$\frac{\ q-q_h\ _{\infty}}{\ q\ _{\infty}}$	<b>farfield</b>	<b>nearfield</b>	<b>nearfield/</b>	<b>GMRES</b>
<b>(s)</b>					<b>(s)</b>	<b>(s)</b>	<b>farfield</b>	<b>iterations</b>
169	2	1.000E-5	2.543E-1	9.821E-1	0.664	1.611	2.426	11
161	3	1.000E-5	4.493E-2	3.608E-1	0.673	1.626	2.415	9
157	4	1.000E-5	4.562E-3	3.040E-2	0.701	1.631	2.328	7
157	5	1.000E-5	1.209E-3	5.157E-3	0.743	1.615	2.174	6
157	6	1.000E-5	7.629E-4	2.049E-3	0.789	1.626	2.061	6
159	7	1.000E-5	7.558E-4	2.098E-3	0.804	1.620	2.014	6

Table 4: Black box parameter study: number of Chebyshev nodes

## B. Black box fast multipole method parameter study

To justify the use of the black-box FMM parameters used in the present work a parameter study was conducted to assess the effect on the accuracy of the boundary solution. The two pertinent parameters that were studied include the number of Chebyshev nodes  $n_{ch}$  used for interpolation in the black-box far-field approximation and the precision  $\epsilon$  used to truncate terms during Singular Value Decomposition. The study was performed using the torus geometry shown in Section 5.1 with 2048 degrees of freedom and cubic T-spline basis functions. Identical boundary data to that in Section 5.1 was prescribed.

In the first parameter study a value of  $\epsilon = 10^{-5}$  was fixed while varying  $n_{ch}$  from 2 to 7. The results are shown in Table 4 which demonstrates that a compromise is found between using low and high  $n_{ch}$  values. Low values result in faster farfield computations but at the cost of accuracy that necessitates further GMRES iterations. Higher values lead to slower farfield computations but higher accuracies that reduce the number of GMRES iterations. From this study values of  $n_{ch} = 4, 5, 6$  are recommended which strike a reasonable compromise between speed and accuracy.

The second parameter study used a fixed value of  $n_{ch} = 5$  while varying  $\epsilon$  from  $10^{-1}$  to  $10^{-7}$  to investigate the effect of SVD tolerance on the final boundary element solution. The results for this study are illustrated in Table 5 where, as before, a com-



<b>solver</b>					<b>typical</b>	<b>typical</b>	<b>nearfield/</b>	<b>GMRES</b>
<b>runtime</b>	$n_{ch}$	$\epsilon$	$\frac{\ q-q_h\ _{L_2}}{\ q\ _{L_2}}$	$\frac{\ q-q_h\ _{\infty}}{\ q\ _{\infty}}$	<b>farfield</b>	<b>nearfield</b>	<b>farfield</b>	<b>iterations</b>
(s)					(s)	(s)		
175	5	1.000E-1	9.487E-1	3.444	0.694	1.640	2.363	13
163	5	1.000E-2	9.563E-2	2.799E-1	0.723	1.618	2.239	10
158	5	1.000E-3	7.770E-3	3.385E-2	0.696	1.616	2.321	8
158	5	1.000E-4	1.404E-3	6.454E-3	0.710	1.626	2.291	7
157	5	1.000E-5	1.209E-3	5.157E-3	0.743	1.615	2.174	6
156	5	1.000E-6	1.208E-3	5.204E-3	0.780	1.626	2.086	6
160	5	1.000E-7	1.208E-3	5.205E-3	0.835	1.634	1.956	6

Table 5: Black box parameter study: SVD precision

promise must be reached in this case between low and high SVD tolerance values. Low tolerances lead to fast farfield computations but at the cost of a lower solution accuracy and further GMRES iterations. High tolerances lead to longer farfield computations but more accurate solutions. Based on the results of this study a value of  $\epsilon = 10^{-5}$  to  $10^{-6}$  is recommended.

### C. Point charges coordinates for molecular model

The set of coordinates  $\{\mathbf{x}_A^c\}_{A=1}^9$  used to define point charges in the boundary value problem posed over the molecular model in Section 5.2 are defined in Table 6. The bounding box of the molecule geometry is defined by  $(x, y, z) \in [-15.5, 15.5] \times [-15.2, 15.2] \times [-14.9, 14.9]$ .

### References

- [1] Auricchio, F., Beirao da Veiga, L., Buffa, A., Lovadina, C., Reali, A., Sangalli, G., 2007. A fully “locking-free” isogeometric approach for plane linear elasticity problems: a stream function formulation. Computer methods in applied mechanics and engineering 197 (1), 160–172.

$\mathbf{x}_A^c$	$(x_A^c, y_A^c, z_A^c)$
$\mathbf{x}_1^c$	(0.0, 0.0, 0.0)
$\mathbf{x}_2^c$	(-6.2, -6.08, -5.96)
$\mathbf{x}_3^c$	(6.2, -6.08, -5.96)
$\mathbf{x}_4^c$	(6.2, 6.08, -5.96)
$\mathbf{x}_5^c$	(-6.2, 6.08, -5.96)
$\mathbf{x}_6^c$	(-6.2, -6.08, 5.96)
$\mathbf{x}_7^c$	(6.2, -6.08, 5.96)
$\mathbf{x}_8^c$	(6.2, 6.08, 5.96)
$\mathbf{x}_9^c$	(-6.2, 6.08, 5.96)

Table 6: Coordinates of point charges used to define molecular model boundary conditions

- [2] Bazilevs, Y., Calo, V. M., Cottrell, J. A., Evans, J. A., Hughes, T. J. R., Lipton, S., Scott, M. A., Sederberg, T. W., 2010. Isogeometric analysis using T-splines. *Computer Methods in Applied Mechanics and Engineering* 199 (5–8), 229–263.
- [3] Bazilevs, Y., Calo, V. M., Hughes, T. J. R., Zhang, Y., 2008. Isogeometric fluid-structure interaction: theory, algorithms, and computations. *Computational mechanics* 43 (1), 3–37.
- [4] Beatson, R., Greengard, L., 1997. A short course on fast multipole methods. In: *Wavelets, Multilevel Methods and Elliptic PDEs*. Vol. 1. pp. 1–37.
- [5] Bebendorf, M., 2000. Approximation of boundary element matrices. *Numerische Mathematik* 86 (4), 565–589.
- [6] Beer, G., Marussig, B., Duenser, C., 2013. Isogeometric boundary element method for the simulation of underground excavations. *Géotechnique Letters* 3, 108–111.
- [7] Belibassakis, K. A., Gerothathis, T. P., Kostas, K. V., Politis, C. G., Kaklis, P. D., Ginnis, A. I., Feurer, C., 2013. A BEM-ISOGEOmetric method for the ship wave-resistance problem. *Journal of Ocean Engineering* 60, 53–67.

- [8] Bremer, J., Gillman, A., Martinsson, P. G., 2014. A high-order accurate accelerated direct solver for acoustic scattering from surfaces. BIT Numerical Mathematics, published online July 2014.
- [9] Buffa, A., Cho, D., Sangalli, G., 2010. Linear independence of the T-spline blending functions associated with some particular T-meshes. Computer Methods in Applied Mechanics and Engineering 199 (23-24), 1437–1445.
- [10] Buffa, A., Sangalli, G., Vázquez, R., 2010. Isogeometric analysis in electromagnetics: B-splines approximation. Computer Methods in Applied Mechanics and Engineering 199 (17–20), 1143 – 1152.
- [11] Carrier, J., Greengard, L., Rokhlin, V., 1988. A fast adaptive multipole algorithm for particle simulations. SIAM Journal of Scientific and Statistical Computing 9 (4), 669–686.
- [12] Cheng, H., Greengard, L., Rokhlin, V., 1999. A fast adaptive multipole algorithm in three dimensions. Journal of Computational Physics 155, 468–498.
- [13] Cirak, F., Ortiz, M., Schröder, P., 2000. Subdivision surfaces: a new paradigm for thin-shell finite-element analysis. International Journal for Numerical Methods in Engineering 47 (12), 2039–2072.
- [14] Cirak, F., Scott, M. J., Antonsson, E. K., Ortiz, M., Schröder, P., 2002. Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision. Computer Aided Design 34 (2), 137–148.
- [15] Cottrell, J. A., Reali, A., Bazilevs, Y., Hughes, T. J. R., 2006. Isogeometric analysis of structural vibrations. Computer methods in applied mechanics and engineering 195 (41), 5257–5296.
- [16] Duffy, M. G., 1982. Quadrature over a pyramid or cube of integrands with a singularity at a vertex. SIAM journal on Numerical Analysis 19 (6), 1260–1262.
- [17] Evans, J. A., Hughes, T. J. R., 2013. Isogeometric divergence-conforming B-splines for the steady Navier–Stokes equations. Mathematical Models and Methods in Applied Sciences 23 (08), 1421–1478.

- [18] Fong, W., Darve, E., 2009. The black-box fast multipole method. *Journal of Computational Physics* 228 (23), 8712–8725.
- [19] Ginnis, A. I., Kostas, K., Politis, C. G., Kaklis, P. D., Belibassakis, K. A., Gerostathis, T. P., Scott, M. A., Hughes, T. J. R., 2014. Isogeometric boundary-element analysis for the wave-resistance problem using T-splines. *Computer Methods in Applied Mechanics and Engineering* 279, 425–439.
- [20] Greengard, L., Gueyffier, D., Martinsson, P. G., Rokhlin, V., 2009. Fast direct solvers for integral equations in complex three-dimensional domains. *Acta Numerica* 18, 243–275.
- [21] Greengard, L., Rokhlin, V., 1987. A fast algorithm for particle simulations. *Journal of Computational Physics* 73, 325–348.
- [22] Hackbusch, W., 1999. A sparse matrix arithmetic based on H-matrices. Part I: Introduction to H-matrices. *Computing* 62 (2), 89–108.
- [23] Hackbusch, W., Khoromskij, B. N., 2000. A sparse H-matrix arithmetic. *Computing* 64 (1), 21–47.
- [24] Hackbusch, W., Nowak, Z. P., 1989. On the fast matrix multiplication in the boundary element method by panel clustering. *Numerische Mathematik* 54, 463–491.
- [25] Harbrecht, H., Peters, M., 2013. Comparison of fast boundary element methods on parametric surfaces. *Computer Methods in Applied Mechanics and Engineering* 261, 39–55.
- [26] Heltai, L., Arroyo, M., DeSimone, A., 2014. Nonsingular isogeometric boundary element method for stokes flows in 3D. *Computer Methods in Applied Mechanics and Engineering* 268, 514–539.
- [27] Heroux, M. A., Bartlett, R. A., Howle, V. E., Hoekstra, R. J., Hu, J. J., Kolda, T. G., Lehoucq, R. B., Long, K. R., Pawlowski, R. P., Phipps, E. T., Salinger,

- A. G., Thornquist, H. K., Tuminaro, R. S., Willenbring, J. M., Williams, A., Stanley, K. S., 2005. An overview of the trinos project. *ACM Trans. Math. Softw.* 31 (3), 397–423.
- [28] Hughes, T. J. R., Cottrell, J. A., Bazilevs, Y., 2005. Isogeometric analysis: CAD, finite elements, NURBS, exact geometry, and mesh refinement. *Computer Methods in Applied Mechanics and Engineering* 194, 4135–4195.
- [29] Johannessen, K. A., Kvamsdal, T., Dokken, T., 2014. Isogeometric analysis using LR B-splines. *Computer Methods in Applied Mechanics and Engineering* 269, 471–514.
- [30] Kurz, S., Rain, O., Rjasanow, S., 2002. The adaptive cross-approximation technique for the 3-D boundary-element method. *IEEE Transactions on Magnetics* 38 (2), 421–424.
- [31] Li, K., Qian, X., 2011. Isogeometric analysis and shape optimization via boundary integral. *Computer Aided Design* 43 (11), 1427–1437.
- [32] Li, X., Scott, M. A., Sederberg, T. W., 2014. Analysis-suitable T-splines: characterization, refineability, and approximation. *Mathematical Models and Methods in Applied Sciences* 24 (6), 1141–1164.
- [33] Martinsson, P. G., Rokhlin, V., 2007. An accelerated kernel-independent fast multipole method in one dimension. *SIAM Journal on Scientific Computing* 29 (3), 1160–1178.
- [34] Marussig, B., Zechner, J., Beer, G., Fries, T. P., 2014. Fast isogeometric boundary element method based on independent field approximation. *arXiv preprint arXiv:1406.0306*.
- [35] Nguyen-Thanh, N., Kiendl, J., Nguyen-Xuan, H., Wüchner, R., Bletzinger, K. U., Bazilevs, Y., Rabczuk, T., 2011. Rotation free isogeometric thin shell analysis using PHT-splines. *Computer Methods in Applied Mechanics and Engineering* 200 (47), 3410–3424.

- [36] Nishimura, N., 2002. Fast multipole accelerated boundary integral equation methods. *Applied Mechanics Reviews* 55 (4), 299–324.
- [37] Peake, M. J., Trevelyan, J., Coates, G., 2013. Extended isogeometric boundary element method (xibem) for two-dimensional helmholtz problems. *Computer Methods in Applied Mechanics and Engineering* 259, 93–102.
- [38] Peake, M. J., Trevelyan, J., Coates, G., 2015. Extended isogeometric boundary element method (XIBEM) for three-dimensional medium-wave acoustic scattering problems. *Computer Methods in Applied Mechanics and Engineering* 284, 762–780.
- [39] Piegl, L., Tiller, W., 1997. *The NURBS Book*. Springer-Verlag, New York.
- [40] Saad, Y., Schultz, M. H., 1986. Gmres: A generalized minimal residual algorithm for solving nonsymmetric linear systems. *SIAM Journal on scientific and statistical computing* 7 (3), 856–869.
- [41] Sauter, S. A., Schwab, C., 2011. *Boundary element methods*. Springer.
- [42] Schillinger, D., Dedé, L., Scott, M. A., Evans, J. A., Borden, M. J., Rank, E., Hughes, T. J. R., 2012. An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces. *Computer Methods in Applied Mechanics and Engineering* 249-252, 116–150.
- [43] Scott, M. A., Borden, M. J., Verhoosel, C. V., Sederberg, T. W., Hughes, T. J. R., 2011. Isogeometric Finite Element Data Structures based on Bézier Extraction of T-splines. *International Journal for Numerical Methods in Engineering*, 88, 126 – 156.
- [44] Scott, M. A., Hughes, T. J. R., Sederberg, T. W., Sederberg, M. T., 2013. An integrated approach to engineering design and analysis using the autodesk t-spline plugin for rhino3d. *Advances in Engineering Software* (in preparation).

- [45] Scott, M. A., Simpson, R. N., Evans, J. A., Lipton, S., Bordas, S. P. A., Hughes, T. J. R., Sederberg, T. W., 2013. Isogeometric boundary element analysis using unstructured T-splines. *Computer Methods in Applied Mechanics and Engineering* 254, 197–221.
- [46] Simpson, R. N., Bordas, S. P. A., Trevelyan, J., Rabczuk, T., 2012. A two-dimensional isogeometric boundary element method for elastostatic analysis. *Computer Methods in Applied Mechanics and Engineering* 209-212, 87–100.
- [47] Simpson, R. N., Scott, M. A., Taus, M., Thomas, D. C., Lian, H., 2014. Acoustic isogeometric boundary element analysis. *Computer Methods in Applied Mechanics and Engineering* 269, 265–290.
- [48] Takahashi, T., Matsumoto, T., 2012. An application of fast multipole method to isogeometric boundary element method for laplace equation in two dimensions. *Engineering Analysis with Boundary Elements* 36 (12), 1766–1775.
- [49] Vázquez, R., Buffa, A., 2010. Isogeometric analysis for electromagnetic problems. *IEEE Transactions on Magnetics* 46 (8), 3305–3308.
- [50] Wang, P., Xu, J., Deng, J., Chen, F., 2011. Adaptive isogeometric analysis using rational PHT-splines. *Computer-Aided Design* 43 (11), 1438–1448.
- [51] Wang, Y., Benson, D. J., Nagy, A. P., 2015. A multi-patch nonsingular isogeometric boundary element method using trimmed elements. *Computational Mechanics*, 1–19.
- [52] Wei, X., Zhang, Y., Hughes, T. J. R., Scott, M. A., 2015. Truncated hierarchical Catmull–Clark subdivision with local refinement. *Computer Methods in Applied Mechanics and Engineering* 291, 1–20.
- [53] Ying, L., Biros, G., Zorin, D., 2004. A kernel-independent adaptive fast multipole algorithm in two and three dimensions. *Journal of Computational Physics* 196 (2), 591–626.

- [54] Yoon, B., Lenhoff, A., 1990. A boundary element method for molecular electrostatics with electrolyte effects. *Journal of Computational Chemistry* 11 (9), 1080–1086.